Maybe You Should Use Knitr

Ryan Giordano rgiordan@mit.edu

April 9, 2021

Abstract

This is an abstract.

1 Introduction

Knitr is a framework for inserting dynamic R code into documents. I'll focus on using Knitr for LaTeX, but it can be used for other systems too (e.g. markdown). This is not intended to be an introduction to knitr! This is a working example of tools and an organizational framework I've developed over the years with help and advice from all my co-authors for interfacing R code and LaTeX. Even if you don't know knitr, you could probably start to use it by copying this framework, and learning what you need as you go along.

What is knitr? In short, knitr converts an `Rnw` file into a `tex` file. This is done via an R command. Here, for convenience, I wrapped the command in the shell script `knit_to_tex.sh`.

What is in a `Rnw` file? It's basically ordinary TeX, which is essentially unchanged during the knitting process, interleaved with R code chunks, which are delimited with special characters (beginning with `<<...>>=` and ending with `@`). When you knit a document, all the code chunks are executed in order. If you ask it to, the output is printed into the TeX document, either as R code, verbatim, or as graphs. R code can also run silently to prepare for future code chunks, and the arguments to code chunks can refer to the R environment. At this level, knitr is an exteremely flexible way to generate LaTeX dynamically. For example, compare the knitr file `experiment_two.Rnw` (which is what you would write) with the automatically generated `experiment_two.tex`.

If you read knitr tutorials, you might get the idea that you're supposed to write your document as one big knitr file and run your analysis inside your

document. Maybe that works for some people, but it can make compiling your document really slow, and can be extremely annoying when you're working on some part of your document that doesn't really need knitr's functionality. For that reason, I use a setup like the one here. Most of the document is simply in `tex` (see `main.tex`, which would normally contain a lot of theory, all in `tex`), but I `input` the knitted output as `tex` files (in this case, `experiment_one.tex`. and `experiment_two.tex`). Every time you make changes to the experiment files `experiment_*.Rnw` you need to re-knit, of course, but you can ignore knitr when editing the rest of the document. In order to make this work, you need to include (once) some knitr boilerplate, which is contained in `_knitr_header.tex`.

I try to do as little actual analysis as possible in the document itself, again to speed up rendering the paper. Remember that every time you change the text of your `Rnw` file you need to re-knit; if you're solving an optimization problem or even reshaping big dataframes each time you compile your pdf, it will be annoying. For that reason I usually write a preprocessing script that saves all the data I need in a nice format (here, `R_scripts/data/preprocess_data.R`). Then I load that data into knitr and basically just use R to make graphs and define macros (see `load_data.R` and `define_macros.R` scripts).

Going even further, I usually put as little actual R code in the `Rnw` document itself, because debugging in knitr is a pain. Instead, I source `.R` scripts that generate the output I need. Typically I develop these scripts interactively by sourcing them in Rstudio until they look like what I want, and then just use the same code in knitr. Both Sections 3 and 4 here have multiple examples of this.

Finally, there's a ton of fiddly stuff, especially around labeling, floats, and image sizes. Fortunately you only need to figure this out once, and then you can just keep using what you know works. This document contains working examples of the some basic things that you can play with. Feel free to explore and change things on your own or, for some guidance, you can try the exercises in Section 2.

2 Exercises

Some exercises:

- What happens if you use `print` instead of `cat` in the `r_example2` chunk of Section 3?

- What happens if refer to the undefined variable `y` instead of `x` in the `r_example2` chunk of Section 3? How does the error appear?
- What do you seen when you set `knitr_debug <- TRUE` in the `setup` chunk of Section 3?
- Suppose nothing works and you think the `data_path` variable is messed up. How can you print the value of `data_path` in the LaTeX pdf to debug it?
- Add a new TeX macro so you can refer to the standard deviation of x in the text. (Edit the `define_macros.R` file.)
- What do you seen when you set `knitr_cache <- TRUE` in the `setup` chunk of Section 3? (Try knitting before and after removing the `figure` directory.)
- Add a column for ϵ and a row for the median to Table 1.
- Add vertical lines to Table 1.
- Replace the text "Epsilon" with the Greek character in the title of the second ggplot of Figure 2. (Hint: look at the x-axis of Figure 1.)
- Change the printed height and width of the figures by passing arguments to `SetImageSize()`.
- Make the figure lines thicker and text larger by changing the `base_figure_width` variable in `initialize.R`.
- Make the figure lines thicker and text larger by changing the `base_figure_width` variable in `initialize.R`.
- Make two plots side by side that share a common legend using the `GetLegend()` command. The layout should be a horizontal row containing first figure 1, then figure 2, then the shared legend.
- Figure 2 uses `GridExtra::grid_arrange` to make side-by-side images. Can you do the same thing with float environments and two separate images?

3 Experiment One

In between the knitr chunks, this is just an ordinary LaTeX document. The content inside the code chunks gets run in R. By default, the code runs silently. If you add the option `results="asis"` then the output gets inserted verbatim into the tex document. This can be used to make tables or define macros.

$1 + 10 = 11$

$x = 6.000000$

Figure insertion uses a special set of semantics — see below for examples.

I use the `define_macros.R` script to specify macros defined from the `Rdata` file. Examples follow. For this experiment, we generated $1,000$ observations. They looked like a mess, as you can see in Figure 1.

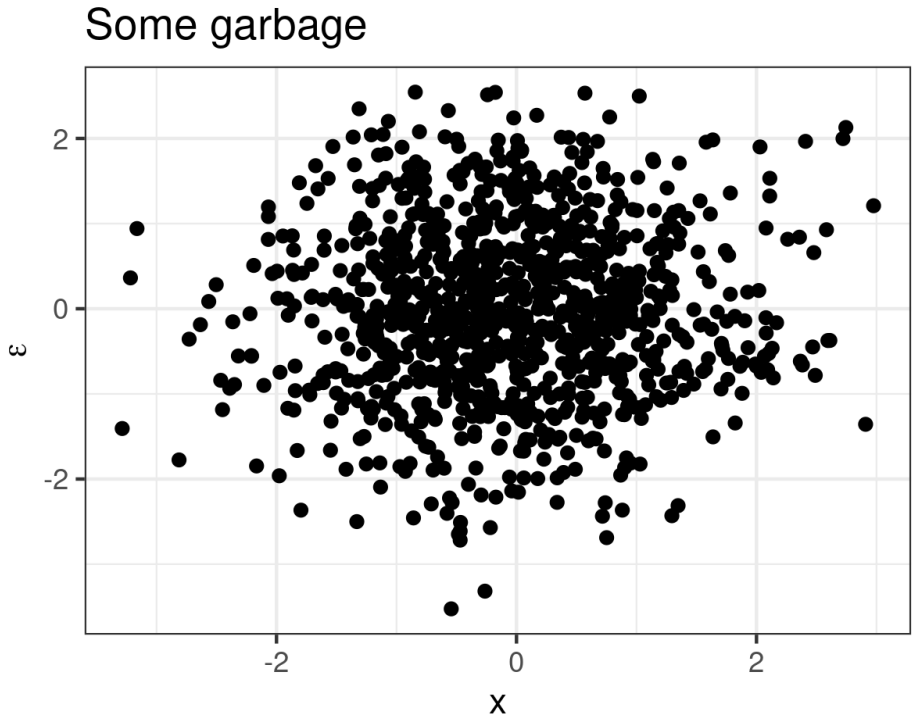

Figure 1: It's nice to have a long figure caption that allows easy access to latex stuff like there were $1,000$ draws of x and ϵ that went into this plot.

And Figure 2 as well. What garbage.

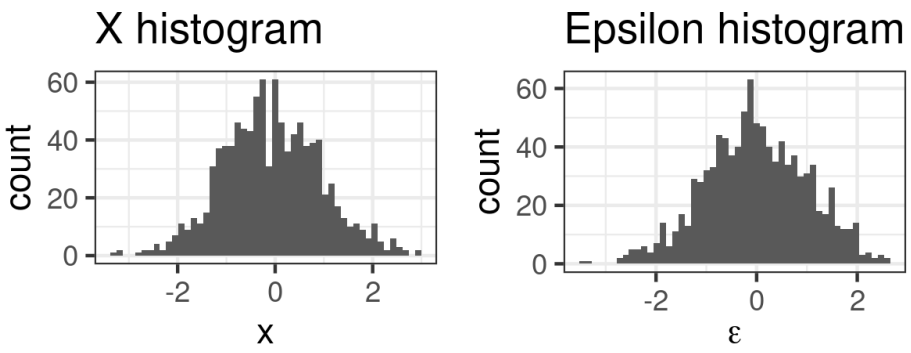

Figure 2: You can reuse this variable for other captions.

4 Experiment Two

We generated $1,000$ observations again, but now we used an offset $\beta = 5$. They looked better, but we already showed you how to make a graph, so instead we'll show it in Table 1.

metric	x	y
mean	-0.0751	-0.3913
sd	1.0112	5.2012
max	2.9757	16.0883

Table 1: Amazing summary stats